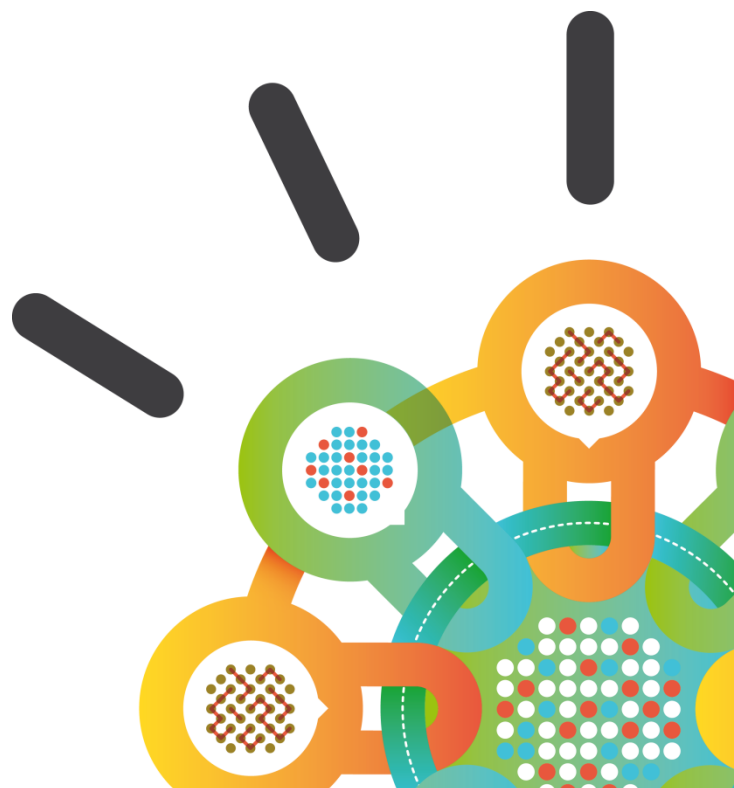




Applikationen im Unternehmen –

ein Einfallstor für Hacker

Stefan.Mandl@de.ibm.com





Der Mythos: “Wir sind sicher”

**Wir haben
Firewalls**

**Wir lassen jedes Jahr ein
Audit durchführen**

**Wir haben SSL
Verschlüsselung**

**Wir haben Netzwerk-
Scanner**





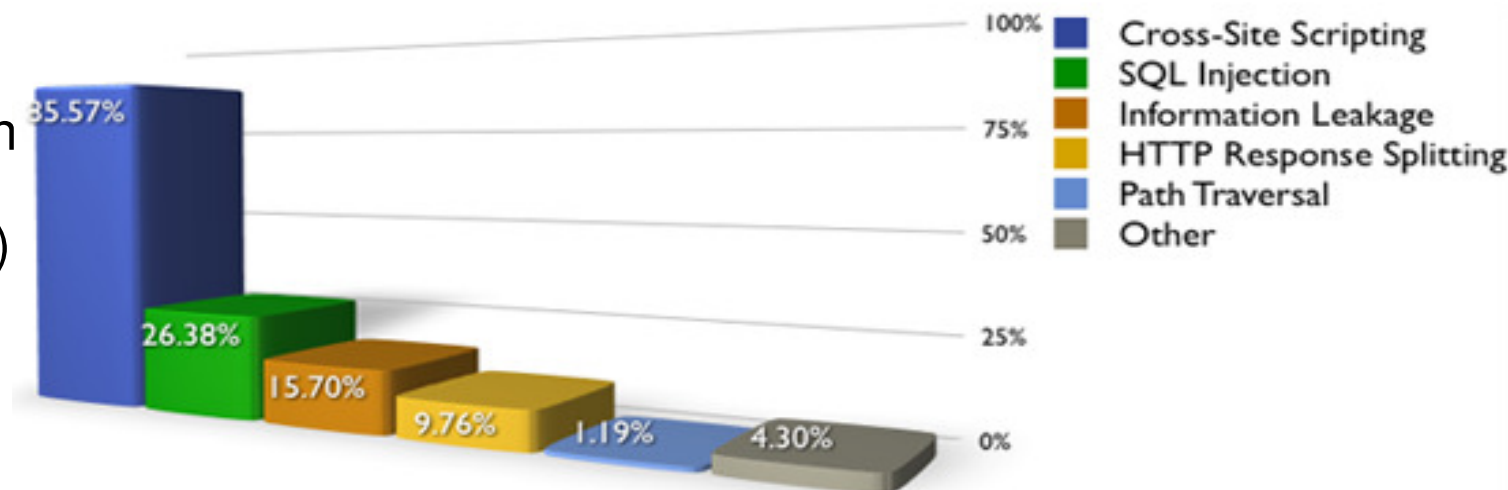
... und die Security Lücken

1. **“Cross-Site Scripting” (XSS)**
2. “Injection Flaws”
3. Verstecktes Ausführen einer Datei
4. Unsichere “Direct Object Reference”
5. Verfälschung eines “Cross-Site Requests”

Top Ten des “Open Web Application Security Project” (OWASP)

6. Informationsverlust und unsaubere Fehlerbehandlung
7. Broken Authentication & Session Management
8. Unsichere Kryptografie Speicherung
9. Unsichere Kommunikation
10. Fehlerhafte Abwehr von URL Zugriffen

Percentage of websites vulnerable by class (Top 5)



Schwachstellen
Statistik
(31.373 Seiten)



Live Hacking

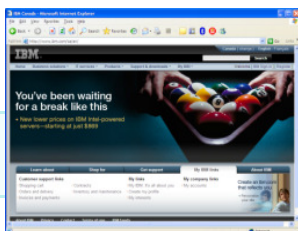
<http://demo.testfire.net/>



Analyse der Applikation

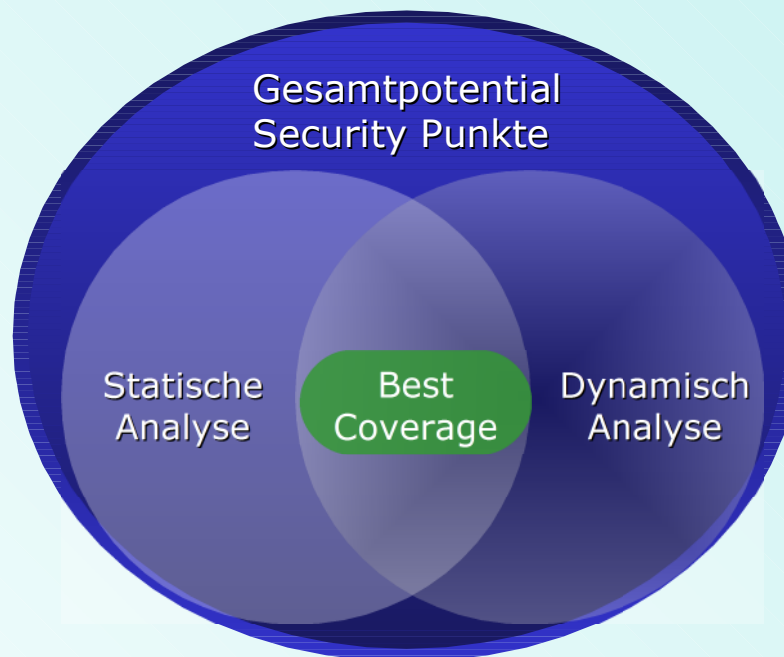
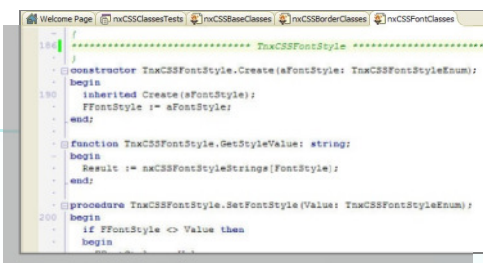
Dynamische Analyse = Blackbox

Scanning nach Security Schwachstellen auf Basis der compilierten Applikation auf Source Code Basis



Statische Code Analyse = Whitebox

Scanning nach Security Schwachstellen auf Source Code Basis





AppScan Standard

AppScan StandardDemo



AppScan Standard Edition

The screenshot displays the IBM Rational AppScan Standard Edition interface. The main window shows a list of 126 security issues for 'My Application', arranged by severity in descending order. The issues include Blind SQL Injection (2), Cross-Site Scripting (8), Database Error Pattern Found (11), DOM Based Cross-Site Scripting (2), HTTP PUT Method Site Defacement (4), Inadequate Account Lockout (1), Permanent Cookie Contains Sensitive Session Information (1), and Predictable Login Credentials (1). The Cross-Site Scripting issue is highlighted, showing a high severity (CVSS 7.5) and a detailed description of the vulnerability. The interface also includes a dashboard with an issue severity gauge, a list of visited URLs, and a status bar at the bottom.

Issue Severity Gauge

Severity	Count
Critical	53
High	32
Medium	24
Low	17

Issue Information: Cross-Site Scripting

- URL:** http://www.altoromutual.com/search.aspx
- Entity:** txtSearch
- Security Risk:** It is possible to steal or manipulate customer session and cookies, which might be used to impersonate a legitimate user, allowing the hacker to view or alter user records, and to perform transactions as that user.

CVSS Metrics... (7.5)

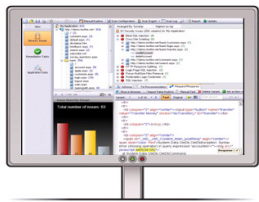
- Base: [Bar]
- Temporal: [Bar]
- Environmental: [Bar]

Dashboard

- Visited URLs: 107/107
- Completed Tests: 22106/22106
- 126 Security Issues
- 53 Critical
- 32 High
- 24 Medium
- 17 Low



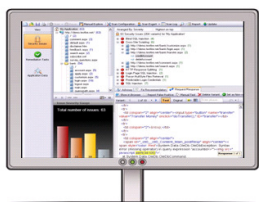
Deployed Application





Deployed Application

Dynamic Analysis



 AppScan Standard
(DAST)



Deployed Application

Dynamic Analysis




AppScan Standard
(DAST)



Deployed Application

Dynamic Analysis

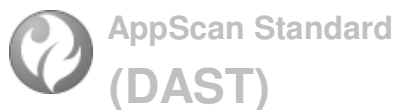


 AppScan Standard
(DAST)



Deployed Application

Dynamic Analysis



Severity	Vulnerabilities		Exceptions		Total
	Type I	Type II	Type I	Type II	
High	17	18	329	0	423
Medium	0	12	11	0	46
Low	1	0	0	0	207

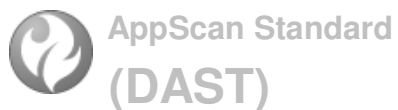
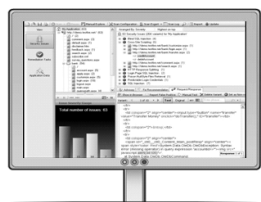
Severity	Type	API Source
High	Injection SQL	rs.java and Statement an
High	Validation Bug	rs.java and Statement an
High	Injection SQL	rs.java and Statement an
High	Injection SQL	rs.java and Statement an
High	Validation Error	rs.java and Statement an
High	Injection SQL	rs.java and Statement an
High	Injection SQL	rs.java and Statement an
High	Validation Error	rs.java and Statement an





Deployed Application

Dynamic Analysis



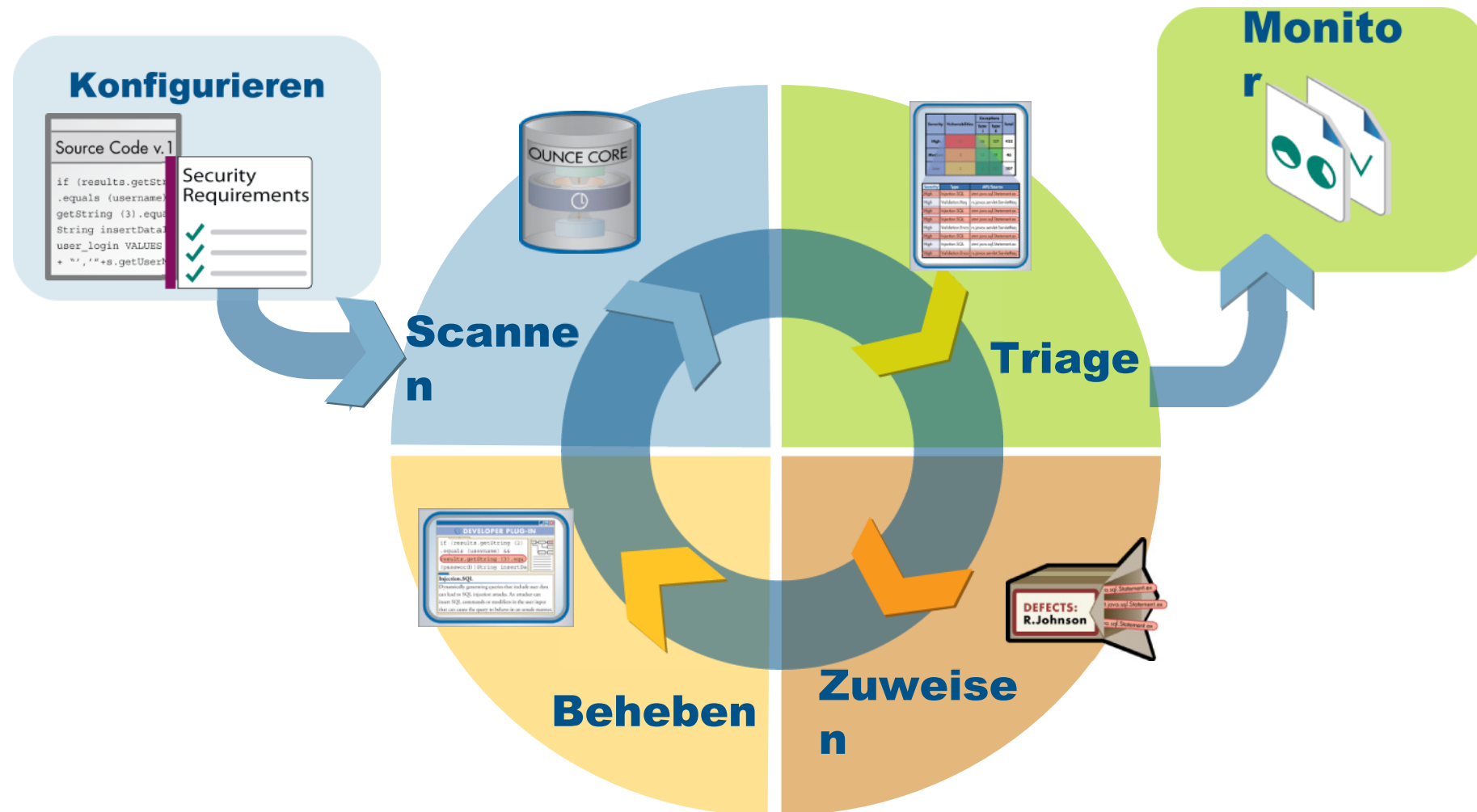
Static Analysis

Severity	Vulnerabilities	Exceptions		Total
		Type I	Type II	
High	25	18	329	433
Medium	2	12	11	46
Low	2			307

Severity	Type	API Source
High	Injection SQL	rs.pjms.sql.Statement an
High	Validation Bug	rs.pjms.sql.Statement an
High	Injection SQL	rs.pjms.sql.Statement an
High	Injection SQL	rs.pjms.sql.Statement an
High	Validation Error	rs.pjms.sql.Statement an
High	Injection SQL	rs.pjms.sql.Statement an
High	Validation Error	rs.pjms.sql.Statement an

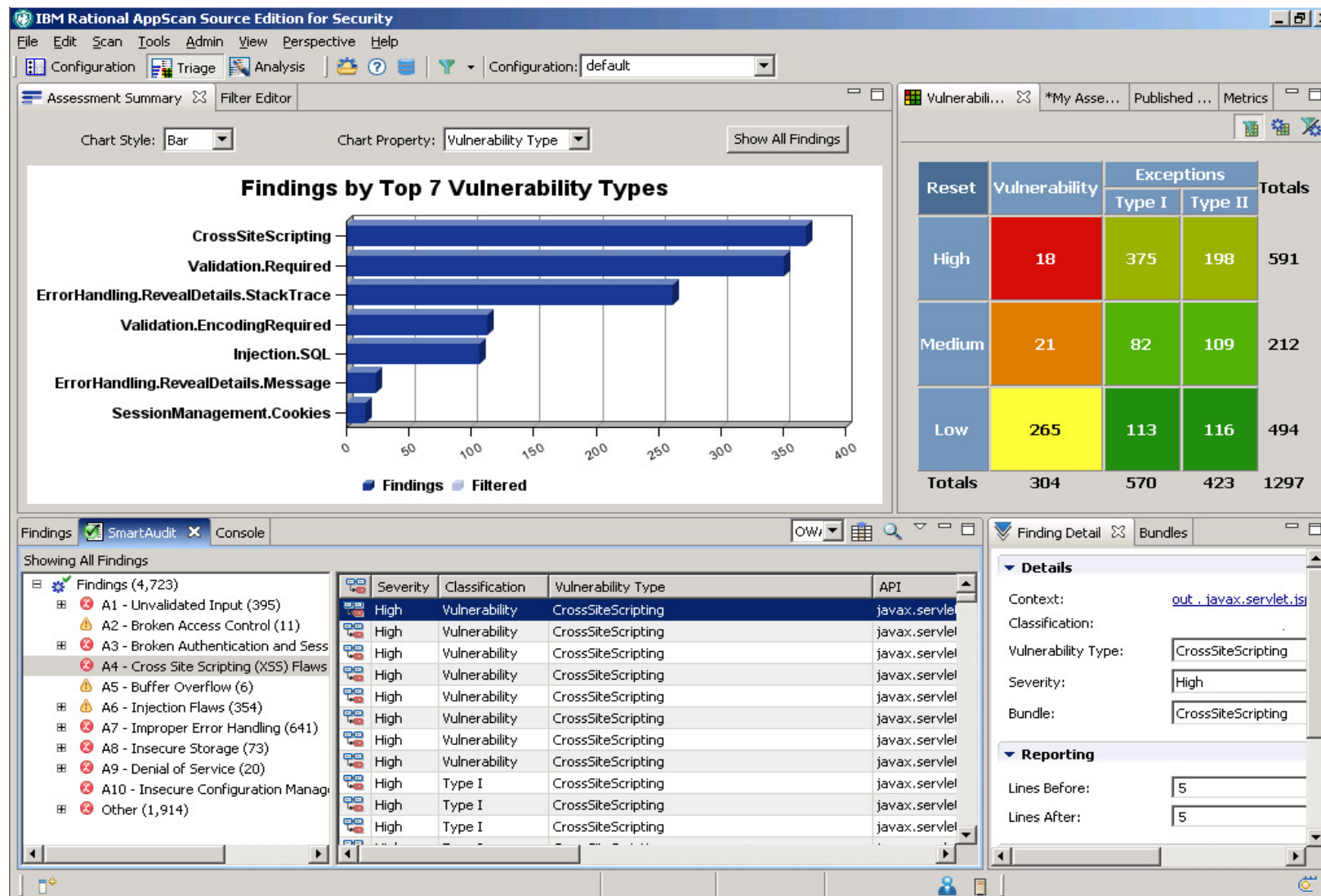


AppScan Source Edition - Workflow





AppScan Source Edition





Virtual Forge CodeProfiler for AppScan Source Edition



- IBM enhanced re-seller agreement with Virtual Forge
 - ▶ Available through Passport Advantage
 - ▶ Support via IBM
- Headquarters in Heidelberg, Germany
- Industry recognized SAP ABAP security experts
- Literally wrote the book on secure SAP ABAP programming
- Gartner – “Cool Vendors in the SAP Ecosystem, 2011”
- Strong relationship & endorsement from SAP

Vulnerability Type

ABAP.AccessControl.AuthCheck.Alias

Alias Authorization in AUTHORITY-CHECK

Authority checks in a SAP system are usually made in the context of the currently logged on user. There are, however, some exceptions from this rule. The command AUTHORITY-CHECK has an option to perform an authority check in the context of an arbitrary user, instead of the currently logged on user. This test case checks for instances where this practice is used and if the identity of the checked user can be controlled by an attacker.

If an attacker can control for which user account an authorization check is performed, he can render the authorization check useless in most instances. This can result in privilege elevation and might become a compliance issue since auditability is impaired. The following risks exist:

- Executing privileged business logic without authorization
- Executing business logic in the name of another user

The command AUTHORITY-CHECK has an optional parameter FOR USER. With this optional parameter, it's possible to perform the authority check in the context of the specified user account, instead of the account of the currently logged on user. Running business logic with different user privileges is dangerous since it impairs accountability. However, this practice may be acceptable under certain circumstances, if a defined technical user account is used. But if the technical user account can be controlled via external input, business functionality might be executed with arbitrary privileges. This is a clear accountability issue.

Example

ABAP

```
myuser = request->get_form_field( 'user' ).  
AUTHORITY-CHECK OBJECT 'S_TCODE'  
FOR USER myuser  
ID 'TCD' FIELD 'SE38'.
```